## REMARKS

Claims 1-32 were presented for examination. The Examiner objected to claim 30 because of an informality. The Examiner rejected claims 1-32 under 35 U.S.C. §101. The Examiner rejected claims 1-3, 5-8, 12-14, 16-22, 26, and 30 under 25 U.S.C. §112, second paragraph. The Examiner rejected claims 1-32 under 35 U.S.C. §102(e) as anticipated by U.S. Patent No. 6,938,247, to Czajkowski (hereafter, "Czajkowski"). No claims have been added. Claims 1, 2, 4, 5, 7-8, 14, 16, 18-20, 22, 26 and 30 have been amended. No new matter has been added. Claims 1 and 23 are independent.

Objection to Claim 30

The Examiner objected to claim 30 because of an informality. The Applicants have amended this claim to overcome the objection.

Rejection of Claims Under 35 U.S.C. §101

The Examiner rejected claims 1-32 under 35 U.S.C. §101, as failing to recite a practical application by producing a physical transformation or producing a useful, concrete and tangible result. The Examiner stated that the claims are useful and concrete but that they fail to produce a tangible result because transformation of data is not a physical transformation and because no result is stored on a non-volatile media or is returned to a user. *See* Office Action, page 3.

Applicants respectfully traverse this rejection. Applicant notes that the claimed invention need not store a result on a non-volatile media for the claims to produce a tangible result. However, claims 1 and 23 do set forth a practical application and result in a computer, which is described in the specification as operating with reduced application compatibility and application sociability problems as a result of using the claimed method and apparatus. *See* Specification, page 8, and claims 1 and 23. Additionally, claim 1 recites the step of responding to a user request for the native resource using the instance of the requested native resource located in the user isolation scope. Therefore, since the claims return a result to a user and produce a tangible result, Applicants respectfully submit that this rejection should be withdrawn.

Rejection of Claims Under 35 U.S.C. §112

The Examiner rejected claims 1-3, 5-8, 12-14, 16-22, 26, and 30 as indefinite for failing to particularly point out and distinctly claim the subject matter that applicants regard as the invention. The Examiner rejects claim 1 as having insufficient antecedent basis for the limitation "the requested resource." Applicants have amended claims 1, 2, 4, 5, 8, 14, 16, 18, 20, and 22 to overcome this rejection.

The Examiner rejected claims 2, 3, 5, 6, 7, 8, 12, 13, 16, 17, 20, and 21 as having insufficient antecedent basis for the limitations referring to steps a, b, c, d, e, and f. Applicants have amended claims 1, 4, 14, 18, 19, and 22, from which claims 2, 3, 5, 6, 7, 8, 12, 13, 16, 17, 20, and 21 depend, to overcome this rejection.

The Examiner rejected claims 7, 8, 14, 18, 19, 22, 26, and 30 as having insufficient antecedent basis. Applicants have amended claims 7, 8, 14, 18, 19, 22, 26, and 30 to overcome these rejections.

Rejection of Claims Under 35 U.S.C. §102(e)

The Examiner rejected claims 1-32 as anticipated by Czajkowski. Independent claim 1 recites:

> A method for isolating access by application programs to native resources provided by an operating system … redirecting to an isolation environment comprising a user isolation scope and an application isolation scope a request for a native resource made by a process executing on behalf of a first user… .

Independent claim 23 recites a similar limitation.

> An isolation environment for isolating access by application programs to native resources provided by an operating system, the isolation environment comprising … a redirector intercepting a request for the native resource made by a process executing on behalf of the user and redirecting the request to the user isolation scope.

Czajkowski does not teach or suggest redirecting, to an isolation environment, a request for a native resource. Czajkowski merely describes a system for inspecting the source code in a Java program, or the Java byte code resulting from an intermediate compilation of a Java program, so that isolated applications may execute in a single virtual machine and may access shared classes. *See* Czajkowski, col. 11, lines 34-57. Czajkowski describes methods for separating out the static fields component of a class, maintaining a separate copy of static fields in shared classes, and creating a class including instance fields corresponding to one or more static fields. *See* Czajkowski, col. 11, lines 16-18 and 60-62; and col. 12, lines 40-45. Czajkowski describes creating a method class for each shared class, the method class including access methods for accessing fields extracted from shared classes. *See* Czajkowski, col. 12, lines 55-57.

Describing a method for transforming the source code in a shared class into source code for three different classes for the purpose of enabling shared access to the class by object-oriented applications instantiating the classes does not teach a method for isolating access to a native resource provided by an operating system. Czajkowski himself provides the following description of classes used in object-oriented programming languages:

> In an object-oriented programming language, data and related methods can be grouped together or encapsulated to form an entity known as an object. All objects in an object-oriented programming system belong to a class, which can be thought of as a category of like objects which describes the characteristics of those objects. Each object is created as an instance of the class by a program. … The class sets out variables and methods for objects which belong to that class. The definition of the class does not itself create any objects. The class may define initial values for its variables, and it normally defines the methods associated with the class … The class may thereby provide all of the program code which will be used by objects in the class, hence maximizing re-use of code which is shared by objects in the class.

*See* Czajkowski, col. 8, lines 19-37. Applicants respectfully submit that this definition describes classes and objects that differ completely from the isolated resources described in the claimed invention.

In contrast to classes, on which Czajkowski is focused, the instant application describes resources provided by the operating system include, for example, a file system and a registry database. *See* Specification, page 1. As described in the specification, file systems provide mechanisms for an application program to open, create, read, copy, modify, and delete data files, and registry databases provide functionality such as storing information regarding hardware physically attached to the computer, how computer memory is set up, various items of application-specific data, and what application programs should be present when the operating system is started. *See* Specification, page 1-2. One of ordinary skill in the art would not consider Java class members to be OS native resources. Czajkowski does not teach native resources provided by an operating system such as file systems or registry databases.

Czajkowski fails to address **inter-process** conflicts over operating system native resource. The pending claims are directed to applications that each have their own process, and addresses the problems that occur because the application processes share the same operating system instance. One of ordinary skill in the art would not find it obvious to solve the problem of inter-process OS native resource clashes based on a description of a solution for intra-process clashes. Even assuming for the sake of argument that native resources are opened using Java function calls to pass in a native resource name (a file path, a registry key name, the name of a named object such as an event or mutex, a COM object GUID, etc.), and that the name is the static field, several problems arise which are not addressed by Czajkowski. First, the names are strings, which are immutable in Java and hence would not need to be divided into per-application copies. Second, even giving each application its own copy of the name does not solve the inter-process conflicts, as each application would attempt to use the same native resource by passing the same name. Third, native resources in Windows can be accessed by traversing a path hierarchy from the root location down to the desired object. Czajkowski does not teach or suggest any mechanism to determine when two applications are using the same instance of a resource when they use different methods to navigate to the resource. For example, one application may open a file using path "C:\Directory1\Directory2\thefile.txt", while another application opens directory "C:\Directory1", and then navigates to subdir "Directory2", then opens the same version of the file - "thefile.txt".

It is not obvious how to extend Czajkowski from handling intra-process data structure clashes to inter-process OS native resource clashes. The Java Virtual Machine does not mediate all access to OS native resources, so it is unable to handle cases where applications access native resources without consulting the JVM (e.g. all native Win32 applications). Therefore, Czajkowski fails to teach isolating access by application programs to native resources provided by an operating system.

Accordingly, Applicants respectfully Czajkowski fails to teach each and every limitation of independent claims 1 and 23. Applicants respectfully request that the Examiner reconsider and withdraw the rejection of independent claims 1 and 23, and dependent claims 2-22 and 24-32.

## CONCLUSION

In view of the above, each of the presently pending claims in this application is believed to be in immediate condition for allowance. Accordingly, the Examiner is respectfully requested to pass this application to issue.

Please charge any additional fees that may be required, or credit any overpayments, to our Deposit Account No. 03-1721.

Respectfully submitted,
CHOATE, HALL & STEWART LLP

/John D. Lanza/
John D. Lanza
Registration No. 40,060

Patent Group
CHOATE, HALL & STEWART LLP
Two International Place
Boston, MA 02110
Tel: (617) 248-5000
Fax: (617) 248-4000
Date: June 11, 2007